A Comparative Study of Four Evolutionary Algorithms for Economic and Economic-Statistical Designs of MEWMA Control Charts

Mahdi Malaki[a], Seyed Taghi Akhavan Niaki[b,*], Mohammad Javad Ershadi[c]

[a]*BSc, Department of Computer Engineering, Sharif University of Technology, Tehran, Iran*

[b]*Professor, Department of Industrial Engineering, Sharif University of Technology, Tehran, Iran*

[c]*MSc, Department of Industrial Engineering, Sharif University of Technology, Tehran, Iran*

Received 5 July, 2011; Revised 11 August, 2011; Accepted 7 December, 2011

Abstract

The multivariate exponentially weighted moving average (MEWMA) control chart is one of the best statistical control chart that are usually used to detect simultaneous small deviations on the mean of more than one cross-correlated quality characteristics. The economic design of MEWMA control charts involves solving a combinatorial optimization model that is composed of a nonlinear cost function and traditional linear constraints. The cost function in this model is a complex nonlinear function that formulates the cost of implementing the MEWMA chart economically. An economically designed MEWMA chart to possess desired statistical properties requires some additional statistical constraints to be an economic-statistical model. In this paper, the efficiency of some major evolutionary algorithms that are employed in economic and economic-statistical design of a MEWMA control chart are discussed comparatively and the results are presented. The investigated evolutionary algorithms are simulated annealing (SA), differential evolution (DE), genetic algorithm (GA), and particle swarm optimization (PSO), which are the most well known algorithms to solve complex combinatorial optimization problems. The major metrics to evaluate the algorithms are (i) the quality of the best solution obtained, (ii) the trends of responses in approaching the optimum value, (iii) the average objective-function-value in all trials, and (iv) the computer processing time to achieve the optimum value. The result of the investigation for the economic design shows that while GA is the most powerful algorithm, PSO is the second to the best, and then DE and SA come to the picture. For economic-statistical design, while PSO is the best and GA is the second to the best, DE and SA have similar performances.

Key Words: Economic-statistical design; Genetic algorithm; Simulated annealing; Particle swarm; Differential evolution.

1. Introduction and Literature Review

Control charts are powerful techniques for maintaining manufacturing processes in desired conditions by graphically plotting observations on the quality characteristics of the product in a suitable manner. For a single quality characteristic of a manufactured product, Shewhart (1931) was the first to introduce an X-bar chart to monitor the mean of the quality characteristic. The exponentially weighted moving average (EWMA) control chart originally introduced by Robert (1959) is employed when a small deviation on the mean of the quality characteristic must be detected. When the vector mean of more than one cross-correlated quality characteristics is to be monitored, the multivariate EWMA (MEWMA) control charts are used.

The economic design of a control chart involves finding the chart parameter values in such a way that the implementation cost of the chart is minimized. Duncan (1956) originally developed a cost model for economic design of an X-bar chart. However, due to the poor statistical performances of the economically designed chart, Duncan's model is not suitable for practical applications. As a result, for the first time, Saniga (1989) developed an economic-statistical model to design an X-bar chart by constraining Duncan's cost model using statistical measures such as type-I and type-II error probabilities. However, since these statistical measures are not applicable for an EWMA-type control chart, Lorenzen and Vance (1986) presented a unified function to model the costs of implementing these charts. The average run length when the process is in-control (ARL_0) and the average run length when the process goes to an out-of-control state (ARL_1) are the two statistical measures that are used in the Lorenzen-Vance function. Later on, Linderman and Love (2000) based on the Lorenzen-Vance function developed economic and economic-statistical models to design the MEWMA

[*] Corresponding author e-mail: Niaki@Sharif.edu

charts. They bounded ARL_0 and ARL_1 to improve the statistical performances of the designed MEWMA charts in an economic-statistical model. In a similar approach, Molnau et al. (2001) by proposing a different method of ARL calculations presented an economic-statistical design for MEWMA chart.

The economic and economic-statistical models that are used to design the MEWMA charts are combinatorial optimization problems with complex and concave cost functions. Furthermore, the statistical constraints of the economic-statistical model make the problem more complicated; requiring a suitable method to solve. Several search-based algorithms in the literature are applied to solve the Lorenzen-Vance-based optimization models of either the economic or the economic-statistical designs of the control charts. Hooke and Jeeves' model (1961) is a pattern search method that was employed by Linderman and Love (2000) to solve the economic-statistical model of a MEWMA control chart. Chou et al. (2006) used a genetic algorithm (GA) approach to solve a similar economic model of variable sampling interval (VSI) EWMA control charts that is based on the Lorenzen-Vance cost function. Nelder-Mead's downhill method (1965) is another search-based algorithm that was used by Serel and Moskowitz (2008) for the Lorenzen-Vance cost function. Serel (2009) concluded that his proposed method does not guarantee convergence to the global optimal solution. Aparisi and Diaz (2004) employed a GA method to design and optimize a MEWMA chart. Niaki et al. (2010) in addition to improving the Lorenzen-Vance cost function presented a GA for the economic and economic-statistical design of the MEWMA control charts.

Evolutionary algorithms that are stochastic and are inspired by natural phenomena evolutionary achieve the optimum value of a complex functions. The stochastic property of these algorithms is the main feature that is helpful to avoid falling in the local optima. The four main traditional evolutionary algorithms that are employed in this paper are genetic algorithm (GA), simulated annealing (SA), particle swarm optimization (PSO), and differential evolution (DE).

GA is a global-search optimization technique that is inspired by biological systems (Davis 1956, Goldberg 1989). In this algorithm, due to many concurrent feasible solution evaluations using an appropriate fitness function, there is a low chance to entrap to local optima. As a result, there are many applications of this algorithm in the literature in different areas such as production planning and control (Disney et al. 2000) and job-shop scheduling problems (Kim et al. 2003).

The PSO algorithm, first developed by Kennedy and Eberhart 1995), is a population- based algorithm inspired by social behavior of swarms in finding particles (for more information see Kennedy et al. 2001 and Omran 2005). Individual swarms that are named particles follow previous particles that were closer to the food source than the other particles. In PSO algorithm, particles fly through the solution space following the current optimum particles.

Another algorithm that is used in this paper is SA that has its origin in the annealing process of metals (Cerny 1985 and Kirkpatrick et al. 1983). Annealing is the process of cooling the molten metal slowly at high temperature to shape crystals. The type of crystals and their energy are mainly the function of cooling rate and other characteristics of metals. In high temperatures, the movements of atoms are fast and as the temperature is reduced, these movements are declined and the atoms start to form crystals with a higher energy than their initial energy. In other words, if the crystallization process takes place at a fast rate, the polycrystalline with a higher energy is made. Thus, to achieve a minimum energy with atoms that have been arranged uniformly in a state called crystalline, a slow cooling rate is necessary. Analogously, SA simulates this process on the optimization models and uses a model called Boltzmann probability distribution in which at each temperature T the probability of the energy of the molten metal is calculated by $P(E) = e^{(-E/k_bT)}$ where E is the size of the system energy and k_b is the Boltzmann constant. In order to implement the Boltzmann function in the minimization process, Metropolis suggested a SA method (For a practical implementation see Vasan and Komaragiri 2009).

The last implemented evolutionary algorithm of this research is the differential evolution (DE) with the main idea of weighting the difference between two vectors to obtain a new vector. Then, the new vector is combined with the basic vector to evolutionary improving the process to achieve the best vector. This algorithm, originally developed by Storn and Price (1997), is mainly used to minimize nonlinear and non-differentiable continuous functions.

The aim of this research is to compare the efficiency of the four aforementioned evolutionary algorithms in solving traditional economic and economic-statistical models of the MEWMA control chart introduced by Linderman and Love (2000) using some performance measures.

The remainder of the paper is organized as follows. The economic and economic-statistical model of designing a MEWMA chart is detailed in the next Section. Section 3 explains the four evolutionary algorithms that are used for the comparison study. The results of investigations on the efficiency of the algorithms and the descriptions of the employed metrics are given in Section 4. Section 5 contains conclusions and some considerations for future research.

2. A Model to Design MEWMA Charts

An economic-statistical model to design a MEWMA chart with minimum implementation cost and desired statistical properties is given in Equation (1), where C denotes the cost function (the Lorenzen-Vance function), n is the sample size, h is the time between two consecutive samples, r is the chart smoothing parameter, and L is the chart control limit parameter (Linderman and Love 2000).

Minimize $C(n,h,r,L)$
Subject to

$$\begin{aligned} &ARL_0 > ARL_L \\ &ARL_1 < ARL_U \\ &n \text{ is a positive integer} \\ &h \text{ and } L > 0 \\ &0 < r \leq 1 \end{aligned} \tag{1}$$

Furthermore, without the first two statistical constraints, a model for designing the chart economically is obtained as given in (2).

Minimize $C(n,h,r,L)$
Subject to

$$\begin{aligned} &n \text{ is a positive integer} \\ &h \text{ and } L > 0 \\ &0 < r \leq 1 \end{aligned} \tag{2}$$

The MEWMA chart along with the Lorenzen-Vance cost function is briefly introduced in Subsections 2-1 and 2-2, respectively.

2.1. Brief background of the MEWMA control chart

In a univariate EWMA chart, first introduced by Roberts (1959), let x_i be the value of a quality characteristic x that must be monitored and without loss of generality consider it as a deviation from its mean μ. The chart depicts z_i given in equation (3) when the on-target mean and variance of x are 0 and σ^2, respectively.

$$z_i = rx_i + (1-r)z_{i-1} \tag{3}$$

in which $z_0 = 0$ and $0 < r \leq 1$ is a parameter that controls the magnitude of smoothing. Furthermore, the mean and variance of z_i are 0 and

$$\sigma_{z_i}^2 = \left\{\frac{r[1-(1-r)^{2i}]}{2-r}\right\}\sigma^2 \tag{4}$$

In order to monitor the means of more than one cross-correlated characteristics, consider $\boldsymbol{X}$ a p-dimensional vector containing the p characteristics with a multivariate normal distribution $N_p(\mu, \boldsymbol{\Sigma})$. Let $\boldsymbol{X_i}$; $i = 1,2,\dots,n$ be the realization on $\boldsymbol{X}$ in which $\boldsymbol{\Sigma}$ is a known covariance matrix. Then the MEWMA chart signals if $Q_i = \boldsymbol{Z}_i'\boldsymbol{\Sigma}_{\boldsymbol{z}_i}^{-1}\boldsymbol{Z}_i > L$, where L and r are the chart parameters that are determined to achieve desired ARL_0 and $\boldsymbol{Z_i}$ is obtained using a multivariate version of equation (3), i.e., $\boldsymbol{Z_i} = r\boldsymbol{X_i} + (1-r)\boldsymbol{Z_{i-1}}$. Furthermore, the asymptotic form of the covariance matrix in this case is as follows.

$$\boldsymbol{\Sigma_Z} = Lim_{i\to\infty}\boldsymbol{\Sigma}_{\boldsymbol{Z}_i} = \left(\frac{r}{2-r}\right)\boldsymbol{\Sigma} \tag{5}$$

Runger and Prabhu (1996) showed that by using the two transformations $b = (2-r)/r$ and $\boldsymbol{\Sigma}^{-1/2}\boldsymbol{X}$, $\boldsymbol{X}$ would have an identity covariance matrix and $Q_i = b \parallel \boldsymbol{Z_i} \parallel^2$. Hence, by setting $UCL = b^{1/2}L^{1/2}$ the statistic $q_t = \parallel \boldsymbol{Z_t} \parallel$ can be plotted as an indicator of the distance in a p-dimensional space.

A parameter that is a metric for the shift of the process mean is the non-centrality parameter δ that is obtained using equation (6). Lowry et al. (1992) showed that the performance of a MEWMA control chart is only a function of this parameter.

$$\delta = (\mu'\boldsymbol{\Sigma}^{-1}\mu)^2 \tag{6}$$

In the off target case, Runger and Prabhu (1996) showed that the performance of a MEWMA chart could be achieved by defining $\delta\boldsymbol{e}$, where $\boldsymbol{e}$ is a p-dimensional vector with a 1 in its first coordinate.

In the next subsection, a function to model the total implementation cost of a MEWMA chart is presented.

2.2. The Lorenzen-Vance cost function

Equation (7) describes a cost function that is used to formulate the cost of implementing a MEWMA chart. In this function, it is assumed that the process starts in an in-control state and the time the process goes to an out-of-control state follows an exponential distribution with mean $1/\theta$.

$$C(n,h,r,L) = \frac{\left\{\frac{C_0}{\theta} + C_1(-\tau + nE + h(ARL_1) + \gamma_1 T_1 + \gamma_2 T_2) + \frac{SF}{ARL_0} + W\right\}}{\left\{\frac{1}{\theta} + \frac{(1-\gamma_1)sT_0}{ARL_0} - \tau + nE + h(ARL_1) + T_1 + T_2\right\}} + \frac{\left\{\left[\frac{a+bn}{h}\right]\left[\frac{1}{\theta} - \tau + nE + h(ARL_1) + \gamma_1 T_1 + \gamma_2 T_2\right]\right\}}{\left\{\frac{1}{\theta} + \frac{(1-\gamma_1)sT_0}{ARL_0} - \tau + nE + h(ARL_1) + T_1 + T_2\right\}} \tag{7}$$

The parameters of the cost function are:

- C_0: The cost per unit time due to nonconformities produced during an in-control state.
- C_1: The cost per unit time due to nonconformities produced during out-of-control states.
- τ: The expected time of occurance of an assignable cause since the previous sample; it is obtained by:

$$\tau = \frac{\int_{jh}^{(j+1)h} e^{-\theta h}(t-jh)dt}{\int_{jh}^{(j+1)h} e^{-\theta h}\,dt} = \frac{1-(1+\theta h)e^{-\theta h}}{\theta(1-e^{-\theta h})} \tag{8}$$

- E: The expected time to sample and chart one item
- T_0: The expected search time when the chart signals a false alarm

- T_1: The expected time to discover an assignable cause
- T_2: The expected time to repair the process
- $\gamma_1 = 0$ if the process is stopped during the search for a cause and 1 if the process is continued while search is performed
- $\gamma_2 = 0$ if the process is stopped during the correction or repair and 1 if the process continues while correction or repair is performed
- S: The expected number of samples taken during an in-control state; it is obtained using

$$S = \frac{e^{-\theta h}}{1-e^{-\theta h}} \tag{9}$$

- F: The cost of a false alarm
- W: The cost of locating and repairing a cause
- a: The fixed cost of taking a sample
- b: The variable cost of taking each unit of a sample

In order to estimate the *ARL*s in the cost function, simulation experiments may be employed using the cost function parameters of n, h, r, and L along with an assumption on the value of the non-centrality parameter δ. Nevertheless, in this paper an approach that was used in Linderman (1998) will be employed.

To find the global solution of models (1) and (2), four different powerful search-based algorithms are applied in the next Sections. To do this, the main characteristics of these algorithms are first discussed in Section 3. Then, the solutions along with the weaknesses and strengths of the algorithms are given in Section 4.

3. Description of the Evolutionary Algorithms

As mentioned in Section 2, the four search-based algorithms to solve models (1) and (2) are SA, DE, GA, and PSO that are described in subsections 3-1, 3-2, 3-3, and 3-4, respectively.

3.1. Simulated Annealing (SA)

The SA implementation of this research is based on the version that is described in Corana et al. (1987). The SA is an iterative process that starts with a given point x_0 and generates a sequence of points $x_0, x_1, \ldots, x_i, \ldots$ tending to the global minimum of the objective function. New candidate points are generated around the current point x_i applying random moves along each coordinate direction, in turn. The new coordinate values are uniformly distributed in intervals centered on the corresponding coordinate of x_i. A candidate point x' is accepted or rejected according to the Metropolis criterion defined as:

- If $\Delta f \leq 0$, then accept the new point: $x_i + 1 = x'$
- Else accept the new point with probability $P(\Delta f) = e^{-\Delta f/T}$ where $\Delta f = f(x') - f(x_i)$ and T is a parameter called temperature.

The process starts at "high" temperature. A sequence of points is then generated until "equilibrium" is approached; that is a sequence of point's x_i whose average value of the cost function achieves a stable value as i increases. During this phase, the step vector $\boldsymbol{V}$ is frequently adjusted to better follow the function behavior. The best point reached is recorded as x_{opt}.

After the thermal equilibration, the temperature T is reduced and a new sequence of moves is made starting from x_{opt} until thermal equilibrium is reached again, and so on. The process is stopped at a temperature low enough such that no more useful improvement can be expected. More details regarding SA are available in Deb (1995) and Ingber (1993).

The other parameters that are used in this algorithm are:

- C: The step adjustment parameter that is used to update the step vector $\boldsymbol{V}$
- N_t: The temperature step size that is used for temperature reduction
- N_s: The number of steps before changing step (step window) that is used for step variation
- R_t: The temperature adjustment coefficient that is used to update the current temperature

3.2. Differential Evolution (DE)

DE is a fast and powerful evolution strategy optimization method that was developed by Storn and Price (1997). It differs significantly from other evolutionary algorithms in the sense that the distance and direction of information from the current population is used to guide the search process. DE has been successfully applied to solve optimization problems arising in different practical applications. While there are several variant strategies of DE, the DE/RAND/1/BIN strategy described in Storn and Price (2000) is used in this research for comparison purposes.

DE algorithms consist of three genetic operators: mutation, crossover, and selection. The parameters of the DE algorithms are the population size (PS), the crossover rate (CR) and a constant scale factor F that influences the diversity of the set of mutant vectors. DE uses the differences between the randomly selected vectors (individuals) as the source of random variations from a third vector (individual), referred to as the target vector. Trial solutions are generated by adding weighted difference vectors to the target vector. This process is referred to as the mutation operator where the target vector is mutated. A recombination or crossover step is then applied to produce an offspring that is only accepted if it improves the fitness of the parent individual.

Mutation and crossover operators generate new individuals and selection operator determines suitable individuals with maximum/minimum fitness values. In this way, the population will contain better individuals.

3.2.1. The Mutation Operator

Mutant individuals are generated using Equation (10) as follows.

$$X_{i,g+1} = V_{r_3,g} + F(V_{r_1,g} - V_{r_2,g}) \qquad (10)$$

with randomly chosen indices r_1, r_2, and r_3 in the range $[1, PS]$. The random indices have to be different from each other and from the running index i. The scale factor F has a constant value in the range [0,2] and controls the amplification of the difference vector of randomly chosen individuals.

3.2.2. The Crossover Operator

The new individuals are generated using the following scheme in the crossover procedure:

$$S_{i,g+1}^{j} = \begin{cases} X_{i,g+1} \ , & r < CR \ or \ j = r \\ V_{i,g}^{j} \ , & r \geq CR \end{cases} \ , \ j = 1,2,\dots,n \qquad (11)$$

Where r denotes a uniform random number in range $[0,1]$, $S_{i,g+1}$ represents trial individuals, n is the vector dimension size, and $CR \in [0,1]$ is a user-defined probability of reproduction. Thus, each offspring is a stochastic linear combination of three randomly chosen individuals when $r < CR$; otherwise the offspring inherits directly from the parent. Even when $CR = 0$, at least one of the parameters of the offspring will differ from the parent's (forced by the condition $j = r$).

3.2.3. The Selection Operator

In the selection procedure of each target individual $V_{i,g}$, the fitness value of the trial individual $X_{i,g+1}$ is compared to that of the target individual $V_{i,g}$. The individual with the maximum fitness value is selected for the next generation.

3.3. Genetic Algorithm (GA)

A solution vector in a GA is similar to a chromosome that is composed of different genes, where each gene represents a variable of the GA. For models (1) and (2) a chromosome of the GA of this research is composed of four different genes as $n, h, r,$ and L shown in Figure (1).

It should be mentioned that in the crossover operation of the GA, the chromosomes that are imported to this step are paired and in each pair, two corresponding genes are replaced with each other with probability of fifty percents. Figure (2) shows an example of the crossover operation. Furthermore, every gene of a chromosome that enters the mutation stage is mutated in the size of d obtained by multiplying three main coefficients; (i) the range of the gene whose generation is performed (the ranges for $n, h, r,$ and L are 30,10,1, and 50, respectively); (ii) a random number with the standard normal distribution; and (iii) a constant number 0.01. Moreover, the evaluation function is the Lorenzen-Vance cost function. Note that since the gene-generation of the chromosomes is taken place in feasible ranges, the constraints on the variables that are related to the ranges are automatically satisfied.

Chromosome #1			
n	h	r	L
12	1.34	0.43	5.06

Chromosome #2			
n	h	r	L
7	6.09	0.98	8.24

Fig. 1. Examples of two chromosomes in the GA

Parent #1			
n	h	r	L
12	1.34	0.43	5.06

Parent # 2			
n	h	r	L
7	6.09	0.98	8.24

Random number generation for each gene

RN#1	RN#2	RN#3	RN#4
0.12	0.63	0.89	0.39

Genes 1 and 4 are replaced with each other and the others remain unchanged because RN# 1 and 4 are less than 0.5

Children #1			
n	h	r	L
7	1.34	0.43	8.24

Children #2			
n	h	r	L
12	6.09	0.98	5.06

Fig. 2. An example of the GA crossover operation

3.4. Particle Swarm Optimization (PSO)

PSO is a population-based stochastic optimization approach that was first developed by Kennedy and Eberhart (1995). The underlying idea of the heuristic arises from social behavior of bird-flocking or fish-schooling (Kennedy & Eberhart 2001, Kennedy et al. 2001, Omran 2005). In a PSO algorithm, swarm or candidate solutions, referred to as particles, are flown around in a D-dimensional search space. Each particle being attracted towards the best position visited by itself (i.e. its own experience) and the position of the best particle in its neighborhood (i.e. the experience of neighboring particles). Each particle has a position vector ($p[]$) encoding a candidate solution to the problem and a velocity vector $v[]$. Additionally, every particle contains a small memory that stores its own best position seen so far ($pbest[]$) and a global best position ($gbest[]$) obtained through communication with its neighbor particles.

In the initialization step, the position of each particle is chosen randomly. Velocities can be initialized randomly or set to 0. This algorithm executes iteratively and its "continuous" version uses a real-valued multidimensional space as belief space. In an iteration, the positions of all particles are evaluated as problem solutions. If a solution is better than any other solution found so far, then the position is stored in the $pbest[]$ vector and the best position among all positions is stored in the $gbest[]$ vector next. The flight velocity and position of each particle in the next iteration is obtained using:

$$v_i = w \times v_i + \left(c_1 \times rand() \times (pbest[]_i - p_i)\right) + \left(c_2 \times rand() \times (gbest[] - p_i)\right)$$
$$p_i = p_i + v_i \qquad (12)$$

where w is the inertia weight (Shi and Eberhart 1998a, 1998b), $rand()$ is a uniform random number on [0,1], c_1 and c_2 are constants called cognitive learning rate and social learning rate, respectively (usually $c_1 = c_2$). The first part of Equation (12) represents the inertia of the pervious velocity; the second part is the "cognition" part, which represents the private thinking by itself; and the third part is the "social" part, which represents the cooperation among the particles. Velocity updates can also be clamped through a user-defined maximum velocity V_{max}, which would prevent them from exploding. As cycle iterates the $gbest[]$-solution converges to the global solution where the stopping criterion is met.

In the original algorithm, particles' velocities on each dimension are restricted to a maximum velocity $V_{\max}$ (a user-defined parameter). If the sum of accelerations on a dimension causes the velocity exceed V_{max}, then the velocity on that dimension is limited to V_{max}. This mechanism prevents the phenomenon of "swarm explosion." In practical applications, there is also a position constriction, where the search space is bounded such that the particles' positions in each dimension are constrained inside the bounds.

4. Performance Comparisons

In this section, the performances of the four algorithms in finding the optimum solution of the economic and economic-statistical models of the MEWMA control chart given in models (1) and (2) are discussed. To do this, the numerical example given in Linderman and Love (2000) is used and the capabilities of the algorithms in finding the optimal solutions for four different values of the non-centrality parameter, δ, are observed. The parameters of the Lorenzen-Vance function in this example are $1/\theta = 100$, $E = 0.05$, $T_0 = 0.0$, $T_1 = 2.0$, $T_2 = 2.0$, $\gamma_1 = \gamma_2 = 1, C_0 = 10.0$, $C_1 = 100.0$, $F = 50.0$, $W = 25.0$, $a = 0.5$, and $b = 0.1$. Furthermore, for performance appraisal of the economic-statistical model two constraints of $ARL_0 \geq 200$ and $ARL_1 \leq 10$ are added to the constraints of the economic model. Linderman and Love (2000) employed the Hooke and Jeeve's method and presented the results that are used in this research to evaluate the power of the four algorithms in finding the optimal solutions. The program coding of the algorithms were performed in Java environment and the codes were run on a 1.86 Giga Hz personal computer with 1-gigabyte ram. Furthermore, each algorithm was run 10 times and the average result was recorded.

The evaluation processes of the algorithms for models (2) and (1) are discussed in Subsections 4-1 and 4-2, respectively. To compare the efficiencies of the algorithms four indicators are used as follows.

1) The quality of the best solution obtained by an algorithm: This metric measures the solution qualities for four different non-centrality parameter values.
2) The convergence path of the search from an initial solution until the stopping criterion; this criterion is also measured for different δs.
3) The computer processing time of the algorithms to reach the known optimum target value: For this indicator the results obtained by Linderman and Love (2000) are used as the optimum target value and the time to reach this target is measured. In situations in which an algorithm does not achieve the target the total processing time of the algorithm is recorded.
4) The average values of all solutions obtained by algorithms: The scatter of solutions that are found in an iteration of the algorithm is measured using this metric. The less the average of the results obtained by an algorithm the better the capability of that algorithm is. This metric is also measured for all δs.

Knowing that parameter values of an algorithm influence both the solution quality and the processing time of reaching the optimum solution, in order to determine the best combination of the parameters of an algorithm, a large number of configurations were first executed. Furthermore, for all algorithms, the initial setting of the parameters was obtained using the commonly recommended set of the previous works (Kennedy & Eberhart 1995, Linderman 1998, Storn & Price 2000, Shi & Eberhart 1998b). Then, the parameter values were changed one by one and the results were considered for the solution quality and speed.

The population size is the most important parameter of the evolutionary algorithms and different population sizes would result in different optimal parameter settings. However, to fairly compare the results of the algorithms, the population size was set 100 for all. Furthermore, all algorithms stop after 500 generations. The parameter values adopted for each of the four EA algorithms are given in Tables (1) to (4).

Table 1
The PSO Parameters

Population size	Maximum velocity	c_1	c_2	Momentum	Neighborhood Size	Termination criteria
100	1	2	2	0.5	20	500

Table 2
The GA Parameters

Population size	Crossover Rate	Mutation rate	Selection rate	Elitism	Termination criteria
100	0.95	0.85	0.5	Yes	500

Table 3
The DE Parameters

Population size	CR	F	Strategy	Elitism	Termination criteria
100	0.95	0.85	E/RAND/1/BIN	Yes	500

The results of the comparison study on the economic and economic-statistical designs of the MEWMA control charts are given in Subsections 4-1 and 4-2, respectively.

Table 4
The SA Parameters

Population size	Starting temperature	N_S	C	N_t	R_t	Termination criteria
100	5	20	2	100	0.85	500

4.1. A Comparative Study for the Economic Design

In this subsection, the performances of the four algorithms in solving the economic design model (2) are measured using the metrics given in subsection 4. Figures (3) to (6) show the quality of the best solution found in the different iterations of the four algorithms for different non-centrality parameter values. The results indicate that while GA outperforms the other algorithms, DE performs second to the best. Then, PSO and SA come to the picture next. Moreover, the PSO and DE have very close performances.

Another criterion to evaluate the algorithms is the progress (convergence path) of each algorithm in achieving the best solution. Figures (7) to (14) show the results of this criterion obtained by the algorithms. In Figures (7) to (10), δ=1.0 and in Figures (11) to (14) δ=3.0. From the figures, it can be seen that GA, PSO, DE and SA are ranked first to fourth. While in Figure (14) and (18), GA reaches a steady-state condition very quickly, for the SA algorithm, the steady-state condition cannot be seen in Figures (7) and (11) until the stopping criterion is taken place.

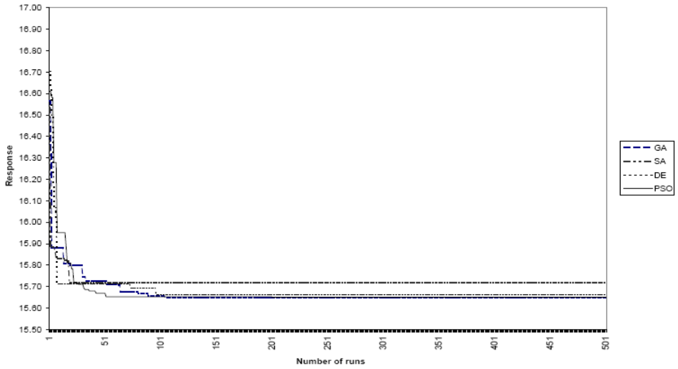


Fig. 3. The quality of the best solution found by each algorithm; δ=3.0

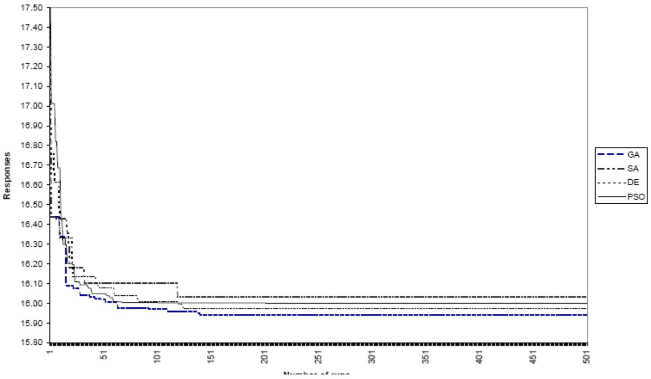
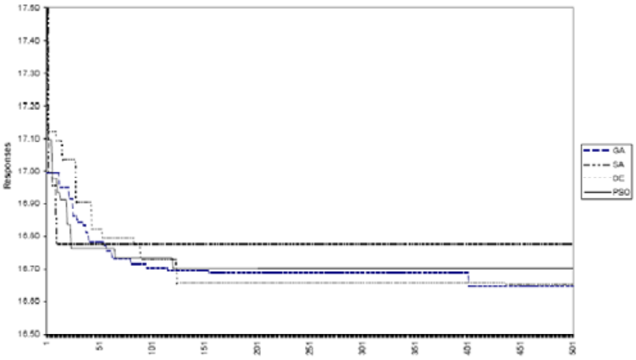


Fig. 4. The quality of the best solution found by each algorithm; δ=2.0



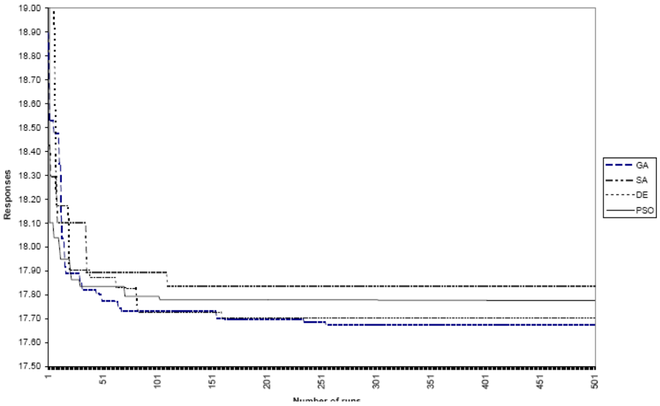


Fig. 5. The quality of the best solution found by each algorithm; δ=1.0

Fig. 6. The quality of the best solution found by each algorithm; δ=0.5

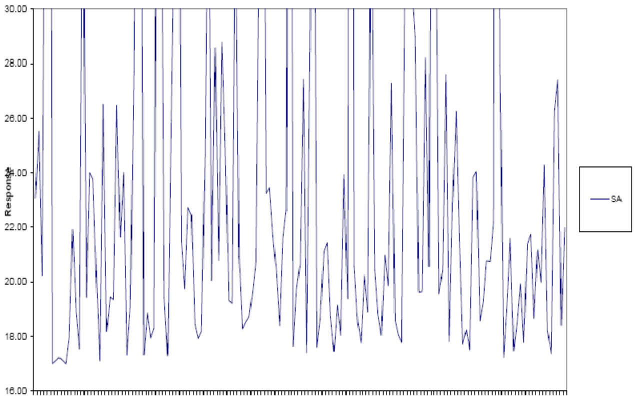


Fig. 7. The convergence path of the SA algorithm to reach the best solution; δ=1.0

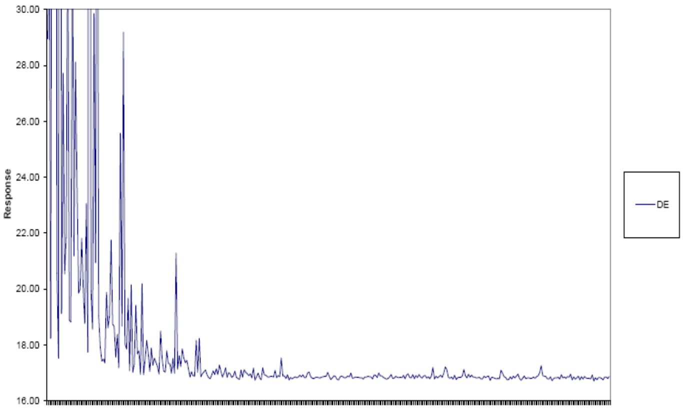


Fig. 8. The convergence path of the DE algorithm to reach the best solution; δ=1.0

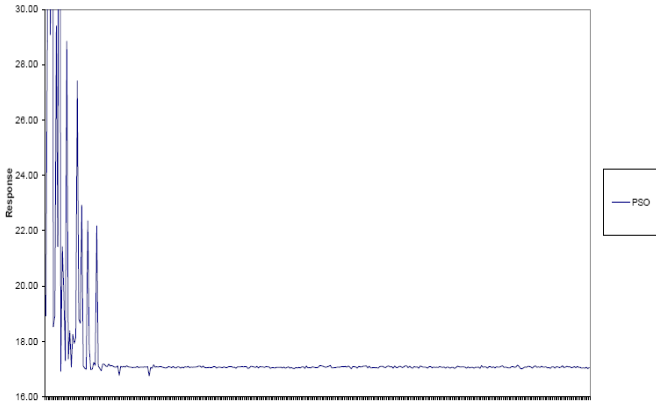


Fig. 9. The convergence path of the PSO algorithm to reach the best solution; δ=1.0

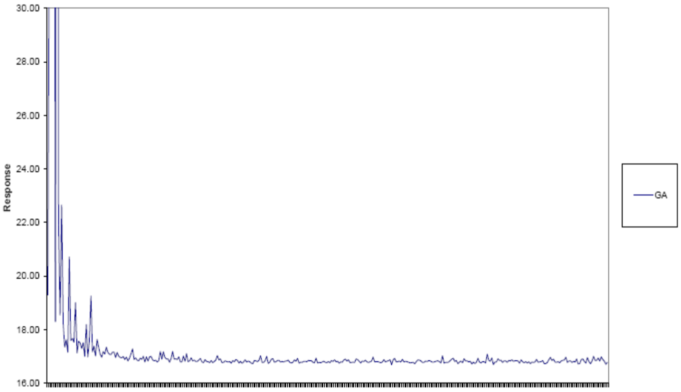


Fig. 10. The convergence path of the GA algorithm to reach the best solution; δ=1.0

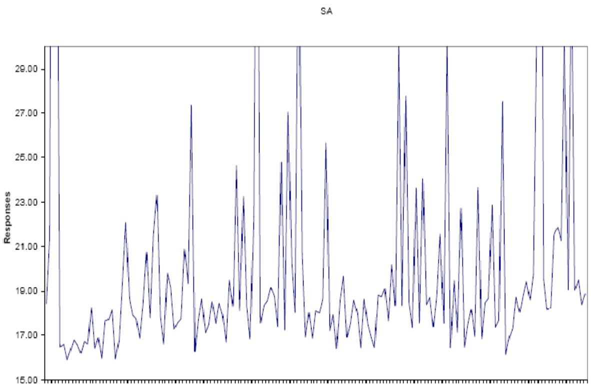


Fig. 11. The convergence path of the SA algorithm to reach the best solution; δ=3.0

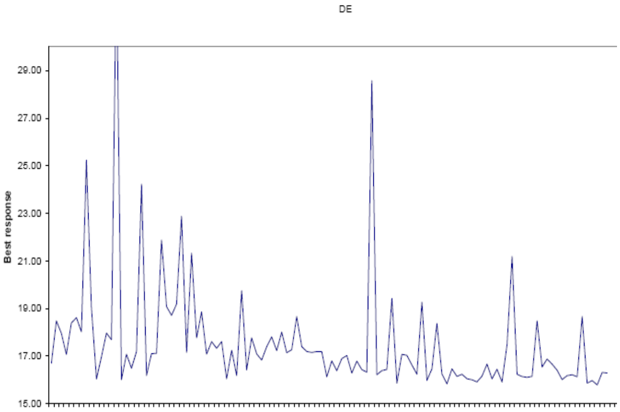


Fig. 12. The convergence of the DE algorithm to reach the best solution; δ=3.0

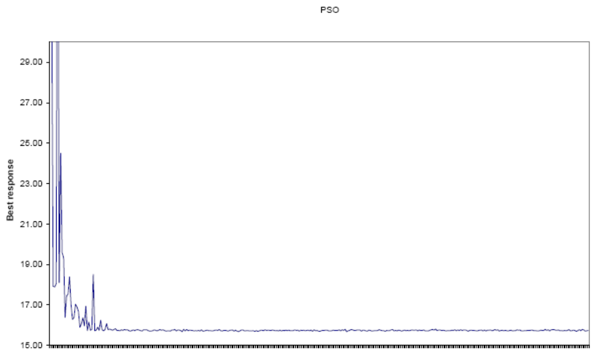


Fig. 13. The convergence path of the PSO algorithm to reach the best δ=3.0

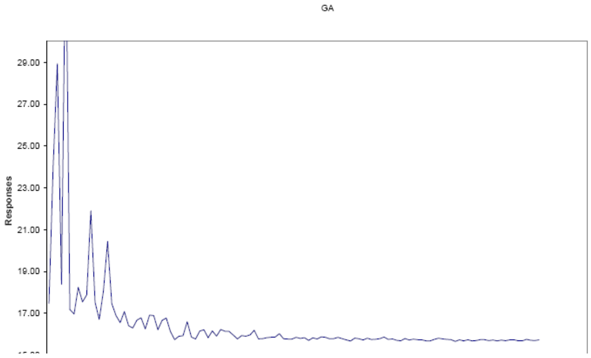


Fig. 14. The convergence path of the GA algorithm to reach the best solution; solution; δ=3.0

The results of the two other major criteria for performance appraisal of the economic design model are shown in Tables (5) and (6) based on different non-centrality parameter values. While the non-centrality parameter values of Table (5) are δ=0.5 and δ=1.0, these values in Table (6) are δ=2.0 and δ=3.0. In these tables, the best solutions found by each algorithm along with the solution obtained by the Hooke and Jeeve's method (H&J) are given. On the one hand, in cases the best-found solutions is more than H&J result, the total computer execution time of the corresponding algorithm is shown. On the other hand, the time to reach the H&J solution is recorded as well.

Table 5
Processing time and average result comparisons; δ=0.5 and 1.0

Algorithm	δ=0.5				δ=1.0			
	Best Sol.	Average Sol.	H&J Sol.	Processing Time (s)	Best Sol.	Average Sol.	H&J Sol.	Processing Time (s)
GA	17.66	18.08	17.80	67.78	16.65	17.16	16.72	244.56
DE	17.70	18.61	17.80	423.19	16.65	18.07	16.72	474.16
SA	17.83	29.02	17.80	898.39	16.78	24.34	16.72	899.94
PSO	17.74	18.25	17.80	138.52	16.70	17.74	16.72	350.97

Table 6
Processing time and average result comparisons; δ=2.0 and 3.0

Algorithm	δ=2.0				δ=3.0			
	Best Sol.	Average Sol.	H&J Sol.	Processing Time (s)	Best Sol.	Average Sol.	H&J Solution	Processing Time (s)
GA	15.94	16.22	15.96	324.66	15.65	16.52	15.65	843.61
DE	15.97	20.12	15.96	899.72	15.66	17.57	15.65	896.94
SA	16.03	20.15	15.96	898.80	15.72	20.30	15.65	902.11
PSO	16.00	17.24	15.96	895.86	15.65	15.93	15.65	210.28

Based on both the processing time and the average solution criteria, the results in Tables (5) and (6) show that GA has the best performances. Then, PSO, DE, and SA are ranked second to fourth, respectively. However, while the best and worst global solutions are obtained by GA and SA, respectively, the global solutions that were found by DE were less than PSO in three cases.

In the next subsection, a similar comparison study is made for the economic-statistical design model in (1).

4.2. A Comparative Study for the Economic-statistical Design

In this section, the four previously described criteria are used for performance assessment of the algorithms used to solve the economic-statistical model. To do this, the same numerical example of Section 4 is employed. Moreover, in order to handle the two augmented constraints of the extended economical-statistical model, the static penalty approach of Hoffmeister and Sprave (1996) that was successfully used in some real-world problems (Schutz and Sprave 1996) is employed to convert the constrained problem into an unconstrained one. Based on this approach, a solution falling outside the bounded region is penalized with a high penalty. The penalty forces the solution to adjust itself in a way that after a few generations/iterations it may fall into the bounded solution space. Mathematically speaking, the penalty function that is applied in this research is:

$$fitness(x) = f(x) \pm \sqrt{H(ARL_0 - ARL_L)^2 + H(ARL_1 - ARL_U)^2}$$

where H is the Heaviside function $H: \Re \to \{0,1\}$ defined as $H(y) = \begin{cases} 1 \ ; & y > 0 \\ 0 \ ; & y \leq 0 \end{cases}$.

Similar to the analysis of subsection 4-1, for different values of the non-centrality parameter Figures (15) to (18) show the quality of the solutions obtained by the evolutionary algorithms under study in different iterations. The results show that while PSO has the best performance in finding the global optimum, DE and SA have roughly similar performances, and the performance of GA is the worst.

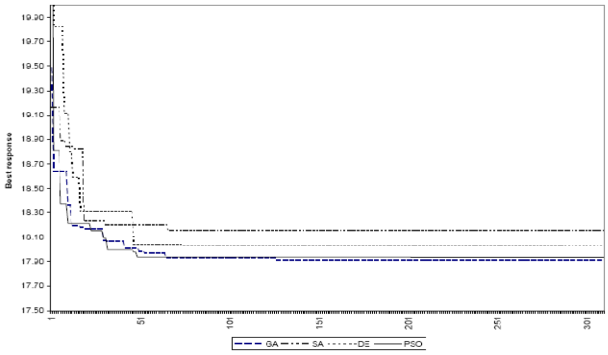


Fig. 15. The quality of the best solution found by each algorithm; δ=0.5

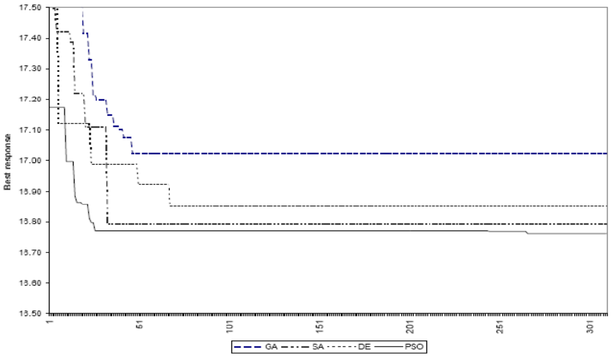


Fig. 16. The quality of the best solution found by each algorithm; δ=1.0

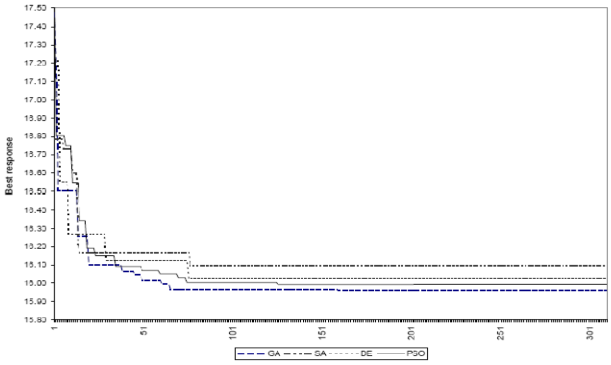


Fig. 17. The quality of the best solution found by each algorithm; δ=2.0

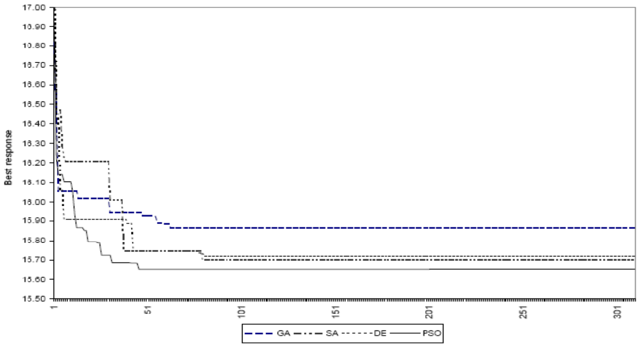


Fig. 18. The quality of the best solution found by each algorithm; δ=3.0

Figures (19) to (22) show the convergence paths of the algorithms in reaching the optimal solutions for non-centrality parameter value of 1.0. Furthermore, Figures (23) to (26) show the path for δ=3.0. As it can be seen from the figures, PSO, GA, DE, and SA are ranked first to fourth.

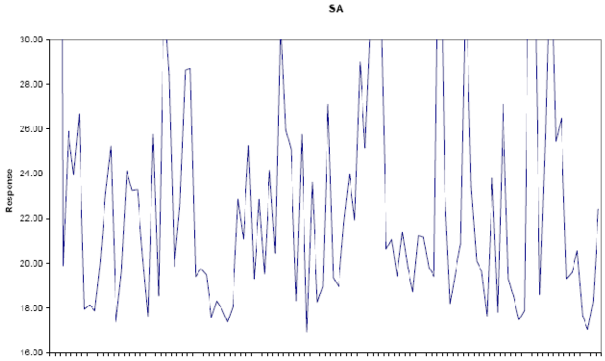


Fig. 19. The convergence path of the SA algorithm to reach the best solution; δ=1.0

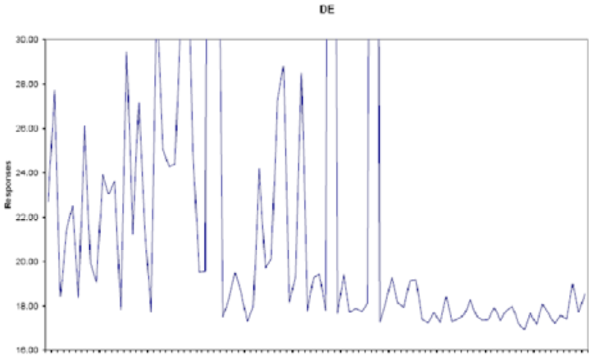


Fig. 20. The convergence path of the DE algorithm to reach the best solution; δ=1.0

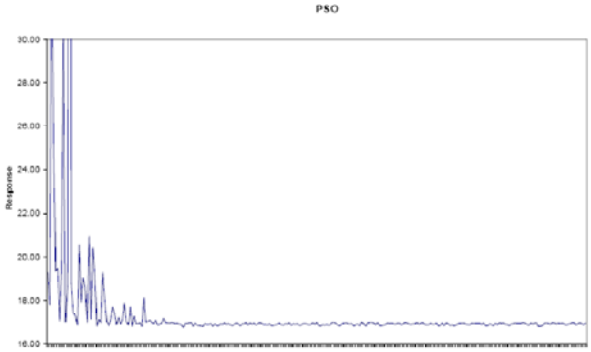


Fig. 21. The convergence path of the PSO algorithm to reach the best solution; δ=1.0

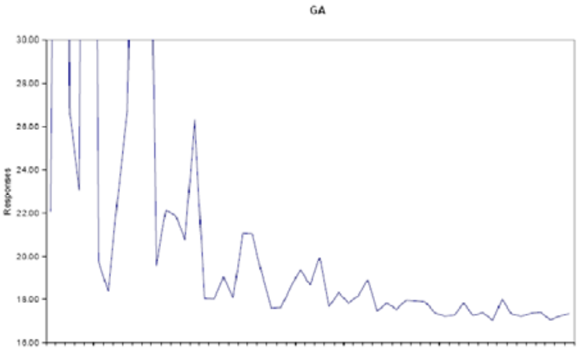


Fig. 22. The convergence path of the GA algorithm to reach the best solution; δ=1.0

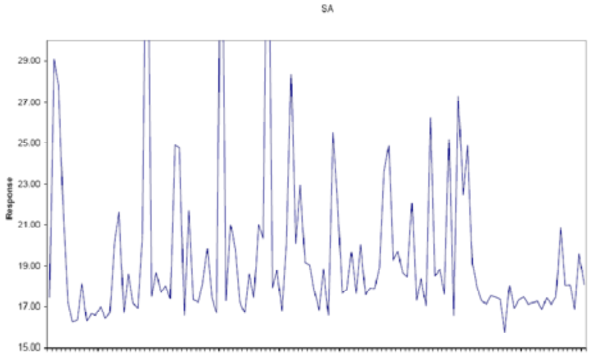


Fig. 23. The convergence path of the SA algorithm to reach the best solution; δ=3.0

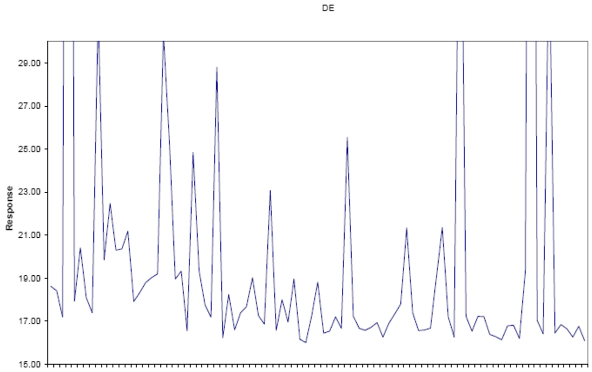


Fig. 24. The convergence path of the DE algorithm to reach the best solution; δ=3.0

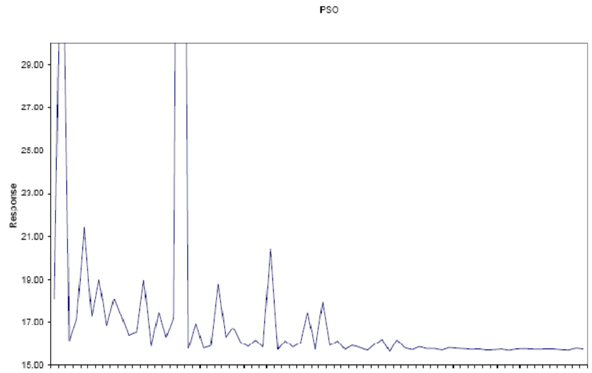


Fig. 25. The convergence path of the PSO algorithm to reach the best solution; δ=3.0

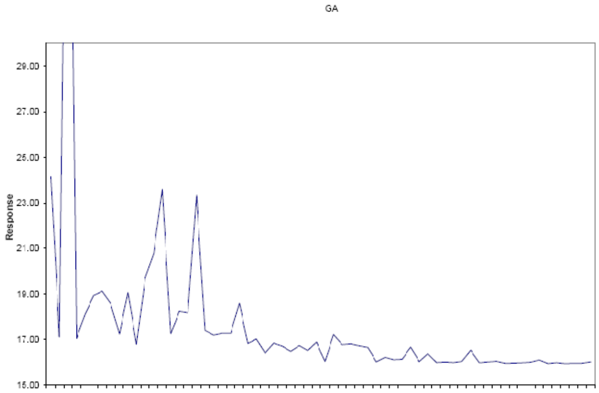


Fig. 26. The convergence path of the GA algorithm to reach the best solution; δ=3.0

Tables (7) and (8) show the average solutions obtained by the algorithms for different non-centrality parameter values. Based on this criterion, while PSO performs the best, GA comes the second to the best, and in almost all of the cases, the SA has the better average solutions than the ones obtained by DE. Moreover, while GA has the least processing time when δ=0.5 and δ=2.0, DE performs the best in case of δ=1.0 and δ=3.0.

Table 7
Processing time and average result comparisons; δ=0.5 and 1.0

Algorithm	δ=0.5				δ=1.0			
	Best Sol.	Avg. Sol.	H&J Sol.	Processing Time (s)	Best Sol.	Avg. Sol.	H&J Sol.	Processing Time (s)
GA	17.91	20.06	17.91	796.81	17.02	22.68	16.77	900.95
DE	18.03	31.71	17.91	896.59	16.85	22.75	16.77	891.24
SA	18.16	27.17	17.91	897.53	16.79	23.41	16.77	897.19
PSO	17.94	18.66	17.91	898.23	16.76	17.47	16.77	899.67

Table 8
Processing time and average result comparisons; δ=2.0 and 3.0

Algorithm	δ=2.0				δ=3.0			
	Best Sol.	Avg. Sol.	H&J Sol.	Processing Time (s)	Best Sol.	Avg. Sol.	H&J Sol.	Processing Time (s)
GA	15.96	16.92	15.97	624.31	15.86	17.69	15.65	892.97
DE	16.03	21.69	15.97	883.92	15.72	20.46	15.65	890.88
SA	16.10	21.50	15.97	897.11	15.70	19.56	15.65	895.05
PSO	15.99	17.18	15.97	894.59	15.65	17.51	15.65	893.19

5. Conclusion and Recommendations for Future Research

In this paper, the economic and economic-statistical models for designing a MEWMA control chart were initially described. Then, four well-known evolutionary algorithms were presented to solve the complex non-linear models. Next, a comparative analysis based on four criteria was used to carry out performance appraisal of the algorithms for different non-centrality parameter values of the chart. The results of the comparison study showed that:

- For the economic design, the GA performed best based on all four criteria. PSO was the second based on the convergence path, processing time and average solution obtained. Meanwhile, DE had the second solution quality. Moreover, SA was the worst algorithm based on all criteria.
- For the economic-statistical design, the PSO ranked first based on all criteria except for the processing time on which GA performed the best. Moreover, GA was the second to the best for the convergence path and average solution criteria. The DE and SA had similar performances for the solution quality and processing time criteria. Furthermore, while DE and SE were the third and fourth algorithms based on the convergence path of the optimal solution, for the average solution criterion the order was converse.

In order to reduce the complexity of the model, only one type of an assignable cause for shifting the mean vector was considered in this research. As a future research, more than one assignable cause are recommended to be considered. Furthermore, the models that involve deviations of the covariance matrix in which unequal sample sizes are used in the cost function may be developed in future research.

6. References

[1] Aparisi, F., Garcia-Diaz, J.C. (2004). Optimization of univariate and multivariate exponentially weighted moving-average Control Charts Using Genetic Algorithms. Computers & Operations Research, 31, 1437-1454.

[2] Cerny, V, (1985). Thermo dynamical approach to the traveling salesman problem: an efficient simulation algorithm. Journal of Optimization Theory and Application, 45, 41-51.

[3] Chou, C.-Y., Chen, C.-H., Liu, H.-R. (2006). Economic design of EWMA charts with variable sampling intervals. Quality and Quantity, 40: 879-896.

[4] Corana, A., Marchesi, M., Martini, C., Ridella, S. (1987). Minimizing multimodal functions of continuous variables with the simulated annealing algorithm. ACM Transactions on Mathematical Software, 13, 262-280.

[5] Davis, L. (1956). Handbook of genetic algorithms. New York, Van Nostrand Reinhold.

[6] Deb, K. (1995). Optimization for engineering design: Algorithms and examples. Prentice-Hall, New Delhi.

[7] Disney, S.M., Naim, M.M., Towill, D.R. (2000). Genetic algorithm optimization of a class of inventory control systems. International Journal of Production Economics, 68, 259-278.

[8] Duncan, A.J. (1956). The economic design of x-charts used to maintain current control of a process. Journal of the American Statistical Association, 51, 228-242.

[9] Goldberg, D.E. (1989). Genetic algorithm in search, optimization, and machine learning reading. Addison-Wesley, MA.

[10] Hoffmeister, F., Sprave, J. (1996). Problem-independent handling of constraints by use of metric penalty functions. in: L.J. Fogel, P.J. Angeline, T. Bäck (Eds.), In Proceedings of the Fifth Annual Conference on Evolutionary Programming (EP'96), MIT Press, San Diego, CA.

[11] Hooke, R., Jeeves, T.A. (1961). Direct search solution of numerical and statistical problems. Journal of the Association for Computing Machinery, 8, 212-229.

[12] Ingber, L. (1993). Simulated annealing: practice versus theory. Mathematical and Computational Modeling, 18, 29–57.

[13] Kennedy, J., Eberhart, R. (1995). Particle swarm optimization.. In Proceedings of IEEE International Conference on Neural Networks IV, Perth, Australia.

[14] Kennedy, J., Eberhart, R., Shi, Y. (2001). Swarm Intelligence. Morgan Kaufmann Publishers, San Francisco.

[15] Kim, Y.K., Park, K., Ko, J. (2003). A symbiotic evolutionary algorithm for the integration of process planning and job shop scheduling. Computers & Operations Research, 30, 1151-1171.

[16] Kirkpatrick Jr. S., Gelatt, C., Vecchi, M. (1983). Optimization by simulated annealing. Science, 220, 498-516.

[17] Linderman, K. (1998). Economic design of multivariate exponentially weighted moving average (MEWMA) control charts. Ph.D. Thesis, Case Western Reserve University.

[18] Linderman, K., Love, T.E. (2000). Economic and economic statistical designs for MEWMA control charts. Journal of Quality Technology, 32, 410-417.

[19] Lorenzen, T.J., Vance, L.C. (1986). The economic design of control charts: A unified approach. Technometrics, 28, 3-10.

[20] Lowry, C.A., Woodall, W.H., Champ, C.W., Rigdon, S.E. (1992). Multivariate exponentially weighted moving average control chart. Technometrics, 34, 46-53.

[21] Molnau, W.E., Montgomery, D.C., Runger, G.C. (2001). Statistically constrained designs of the multivariate exponentially weighted moving average control chart. Quality and Reliability Engineering International, 7, 39-49.

[22] Nelder, J.A., Mead, R. (1965). A Simplex method for function minimization. The Computer Journal, 7, 308-313.

[23] Niaki, S.T.A., Ershadi, M.J., Malaki, M. (2010). Economic and economic-statistical designs of MEWMA control charts- A hybrid Taguchi loss, Markov chain and genetic algorithm approach. International Journal of Advanced Manufacturing Technology, 48, 283-296.

[24] Omran, M. (2005). Particle swarm optimization methods for pattern recognition and image processing. PhD thesis, Department of Computer Science, University of Pretoria, South Africa.

[25] Roberts, G.C. (1959). Control chart tests based on geometric moving averages. Technometrics, 1, 239-250.

[26] Runger, G.C., Prabhu, S.S. (1996). A Markov chain model for the multivariate exponentially weighted moving average control chart. Journal of the American Statistical Association, 91, 1701-1706.

[27] Saniga, E.M. (1989). Economic statistical control chart with an application to X-bar and R charts. Technometrics, 31, 313-320.

[28] Schütz, M., Sprave, J. (1996). Application of partially mixed-integer evolution strategies with mutation rate pooling. in: L.J. Fogel, P.J. Angeline, T. Bäck (Eds.), Proceedings of the Fifth Annual Conference on Evolutionary Programming (EP'96), MIT Press, San Diego, CA.

[29] Serel, D.A., Moskowitz, H. (2008). Joint economic design of EWMA control charts for mean and variance. European Journal of Operational Research, 184, 157-168.

[30] Serel, D.A. (2009). Economic design of EWMA control charts based on loss function. Mathematical and Computer Modeling, 49, 745-759.

[31] Shewhart, W.A.T. (1931). Economic control of quality of manufactured product. Van Nostrand, New York, USA.

[32] Shi, Y., Eberhart, R. (1998a). Parameter selection in particle swarm optimization. In Proceedings of Evolutionary Programming, 98, 591–600.

[33] Shi, Y., Eberhart, R. (1998b). A modified particle swarm optimizer. In Proceedings of the IEEE international conference on evolutionary computation. IEEE Press, Piscataway, NJ.

[34] Storn, R., Price, K. (1997). Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. Journal of Global Optimization, 11, 341-359.

[35] Storn, R., Price, K. (2000). Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. ICSI, TR95-012, Available at: http://icsi.berkely.edu/~storn/literal.html.

[36] Vasan, A., Komaragiri, S.R. (2009). Comparative analysis of simulated annealing, simulated quenching and genetic algorithm for optimal reservoir operation. Applied Soft Computing, 9, 274-281.